

3D/2D modelling suite for integral water solutions

DELFT3D

Deltares systems

NEFIS Viewer Selector

User Manual

Deltares
Enabling Delta Life 

ViewerSelector

Inspection, selection and retrieval of data from a NE-FIS file

User Manual

Hydro-Morphodynamics & Water Quality

Version: 1.23
Revision: 41593

14 December 2015

ViewerSelector, User Manual

Published and printed by:

Deltares
Boussinesqweg 1
2629 HV Delft
P.O. 177
2600 MH Delft
The Netherlands

telephone: +31 88 335 82 73
fax: +31 88 335 85 82
e-mail: info@deltares.nl
www: <https://www.deltares.nl>

For sales contact:

telephone: +31 88 335 81 88
fax: +31 88 335 81 11
e-mail: sales@deltaressystem.nl
www: <http://www.deltaressystem.nl>

For support contact:

telephone: +31 88 335 81 00
fax: +31 88 335 81 11
e-mail: support@deltaressystem.nl
www: <http://www.deltaressystem.nl>

Copyright © 2015 Deltares

All rights reserved. No part of this document may be reproduced in any form by print, photo print, photo copy, microfilm or any other means, without written permission from the publisher: Deltares.

Contents

1	Guide to this manual	1
1.1	Introduction	1
1.2	Changes with respect to previous versions	1
2	Introduction	3
2.1	What is the ViewerSelector?	3
2.2	For whom?	3
2.2.1	Why using the ViewerSelector?	4
3	Getting started	5
3.1	Overview of Delft3D	5
3.2	Starting Delft3D	5
3.3	Getting into ViewerSelector	6
3.4	Exploring some command options	8
3.5	Exiting the ViewerSelector	10
4	General operation	11
4.1	Environment	11
4.2	User Interface	11
4.3	Conventions	11
5	Commands	15
5.1	Help command	15
5.2	NEFIS file command	15
5.3	Display commands	16
5.4	Data selection and manipulation commands	16
5.5	Memory management commands	17
5.6	Write command	17
5.7	Tailor-made process execution commands	17
5.8	Shell escape command	18
5.9	Exit ViewerSelector	18
6	Error messages	19
7	Tutorial	21
7.1	Example 1	21
7.2	Example 2	22
7.3	Example 3	23
7.4	Example 4	24
	References	25
A	Appendix	27
A.1	Distribution and installation notes	27
A.1.1	Environment	27
A.1.2	Starting the ViewerSelector	27
A.2	Command summary	28

DRAFT

List of Figures

3.1	Title window of Delft3D	5
3.2	Main window Delft3D-MENU	6
3.3	Selection window for Utilities	6
3.4	Select working directory window	7
3.5	Select working directory window to set the working directory to <viewer_selector>	7
3.6	Current working directory displayed in title bar	7
3.7	Main window of the ViewerSelector	8
3.8	Overview of ViewerSelector commands	8
3.9	Result from the overview command “disp stat”	9
3.10	Content of the group ‘map-series’	9
3.11	Overview water levels	10

DRAFT

DRAFT

1 Guide to this manual

1.1 Introduction

As the use of NEFIS files increases, the need for a common tool for inspecting these files and selecting data from these files grows.

This need has resulted in the design and implementation of a product named ViewerSelector.

This product provides:

- ◇ Fast inspection of NEFIS files.
- ◇ Selection and retrieval of data from a NEFIS file.
- ◇ Simple data manipulation functions.
- ◇ Output of data to an ASCII-file (TEKAL format).
- ◇ An interface to 'third party' processes (scripts, filters, programs, etc.)

This manual describes how the ViewerSelector works.

Chapter 2: Introduction, provides specifications of the ViewerSelector, why and when you should use the utility

Chapter 3: Getting started, explains the use of the overall menu program, which gives access to all Delft3D modules. Last but not least you will get a first introduction into the ViewerSelector.

Chapter 4: General operation, gives conventions and the general philosophy.

Chapter 5: Commands, explains in detail all commands.

Chapter 6: Error messages, gives an overview of the messages when an error occurs.

Chapter 7: Tutorial, gives examples how to use the ViewerSelector.

References, provides a list of publications and related material.

Appendix A: Appendix Distribution and Installation notes are given.

1.2 Changes with respect to previous versions

Version	Description
1.23	Update of screen graps
1.22	Bookmarks added for PDF version.
1.21	Delft-GPP changed to GPP. Operation of <i>Change working directory</i> updated in Chapter 3. Extension of Delft3D-MENU with Delft3D-QUICKPLOT.
1.20	Reference version for these change notes.

DRAFT

2 Introduction

This chapter gives a short introduction to what the ViewerSelector is, for whom it is meant and why it was developed.

2.1 What is the ViewerSelector?

The ViewerSelector is a tool for inspecting NEFIS files and selecting data from these files. Inspecting a NEFIS file means that you get an overview of the data in this file (meta-data). Selecting data means that you specify which data you want to be read from the NEFIS file to memory.

Selected data can be manipulated with some simple built-in functions. These functions are:

- ◇ add, subtract
- ◇ divide
- ◇ multiply
- ◇ maximum
- ◇ minimum
- ◇ average.

The selected data can be viewed on screen or written to an ASCII-file in so-called TEKAL format.

Of course, the ViewerSelector cannot always manipulate or write the data in the way you want. To prevent the ViewerSelector from having a wide variety of functions for manipulating data and writing it in all kinds of ways, it has an open interface. Through this powerful interface you can start processes with selected data. These processes may also produce new data and return it to the ViewerSelector. For building such processes see the ViewerSelector Programmers Manual.

2.2 For whom?

The ViewerSelector is meant for all users of NEFIS files, i. e. end-users and application programmers.

The ViewerSelector can be used by the end-user who has some knowledge about the way his/her NEFIS files are used and structured. You will experience that, when applying the ViewerSelector in your environment, you often use the same commands in the same order with some minor changes, for example only the filenames or element names. If this is true, it is time for you to embed the ViewerSelector in a script file. How this can be done, is explained in this document.

The ViewerSelector can also be used by application programmers, who maintain applications that produce NEFIS files. Often these programmers provide end-users with tools for post-processing. These application programmers can now embed the ViewerSelector in their post-processing environment, write tailor-made scripts and filters, and so on.

The ViewerSelector is also a helpful tool for programmers, building applications with NEFIS files. They can use it for inspecting the files.

Remark:

- ◇ The command line oriented user interface may need to be upgraded to a state of the art window oriented user interface. For using the ViewerSelector in scripts and filters



however, a command line based version is more appropriate.

2.2.1 Why using the ViewerSelector?

The ViewerSelector is built to provide users of NEFIS files with an easy inspection and data retrieving tool. The application programmers can now put their effort in developing scripts or even more sophisticated user interfaces for post-processing, dedicated to their special environment. They also can built processes to be executed through the open interface of the ViewerSelector. You can think of special print programs, graphics applications, conversion filters, and so on. How this can be done, is explained in the ViewerSelector Programmers Manual [ViewerSelector \(1993\)](#).

DRAFT

3 Getting started

3.1 Overview of Delft3D

The Delft3D program suite is composed of a set of modules (components) each of which covers a certain range of aspects of a research or engineering problem. Each module can be executed independently or in combination with one or more other modules.

Delft3D is provided with a menu shell through which you can access the various modules. In this chapter we will guide you through some of the input screens to get the look-and-feel of the program. In the Tutorial, Chapter 7, you will learn to define a simple scenario.

3.2 Starting Delft3D

To start Delft3D:

- ◇ On an MS Windows platform: select *Delft3D* in the *Programs* menu.
- ◇ On Linux machines: type `delft3d-menu` on the command line.

Next the title window of Delft3D is displayed, [Figure 3.1](#).

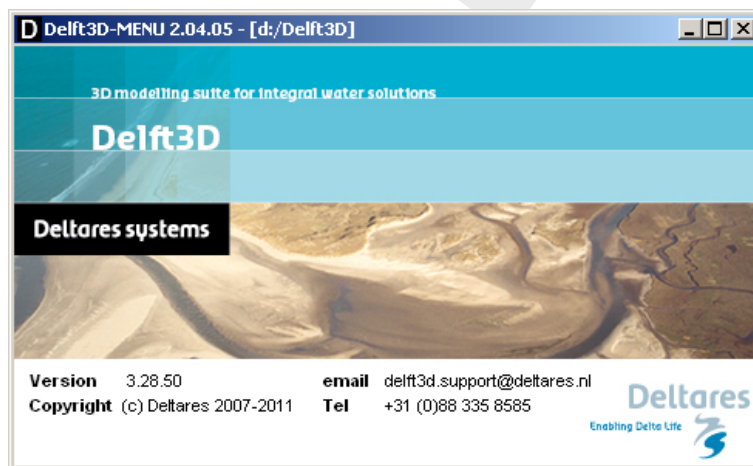


Figure 3.1: Title window of Delft3D

After a short while the main window of the Delft3D-MENU appears, [Figure 3.2](#).

Several menu options are shown. For now, only concentrate on exiting Delft3D-MENU, hence:

- ◇ Click on the *Exit* push button.

The window will be closed and you are back in the Windows Desktop screen for PCs or on the command line for Linux and UNIX workstations.

Remark:

- ◇ In this and the following chapters several windows are shown to illustrate the presentation of Delft3D-MENU and ViewerSelector. These windows are grabbed from the PC-platform. For Linux workstation the content of the windows is the same, but the colours may be different.



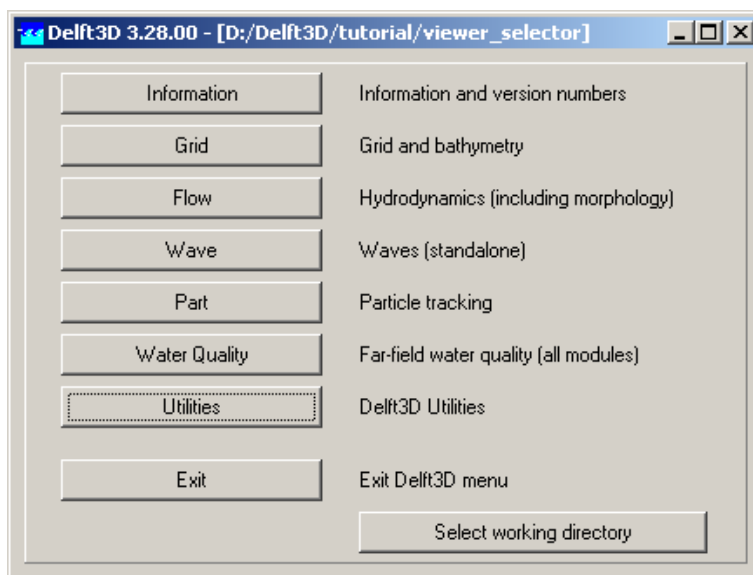


Figure 3.2: Main window Delft3D-MENU

3.3 Getting into ViewerSelector

To continue restart the menu program as indicated above.

- ◇ Click the Utilities button.

Next the selection window for Delft3D Utilities is displayed for post-processing with Delft3D-QUICKPLOT or GPP and to view and select NEFIS files using the ViewerSelector: see [Figure 3.3](#).

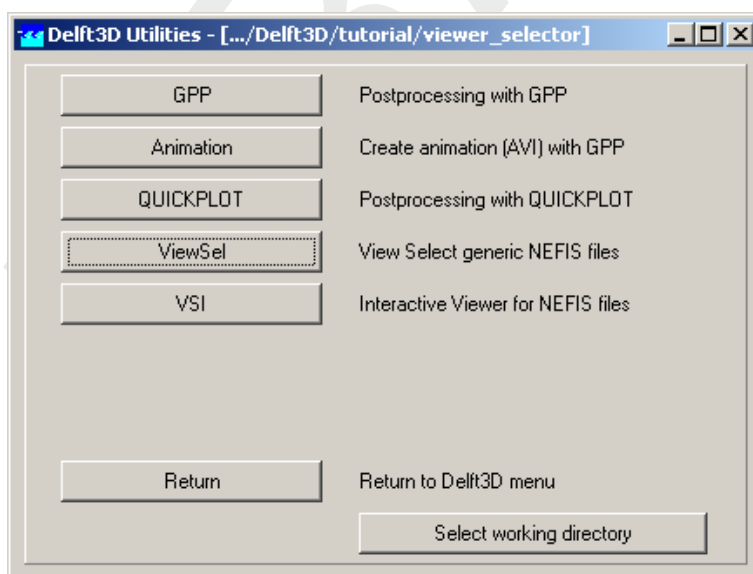


Figure 3.3: Selection window for Utilities



Remark:

- ◇ Post-processing with Delft3D-QUICKPLOT or GPP can also be accessed as part of every module selection window.

Before continuing with the *ViewSel* selection of this **Delft3D Utilities** window, you must select

the directory in which you are going to select and view NEFIS files:

- ◇ Click the *Change working directory* button.

Next the **Select working directory** window, [Figure 3.4](#), is displayed (your current directory may differ, depending on the location of your Delft3D installation).

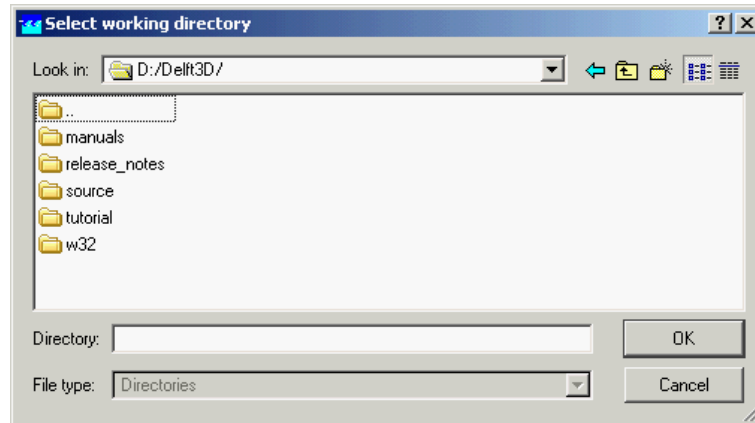


Figure 3.4: *Select working directory window*

- ◇ Navigate to and open the <Tutorial> directory of your Delft3D Home-directory.
- ◇ Select the <viewer_selector> directory and close the **Select working directory** window by clicking **OK**, see [Figure 3.5](#).

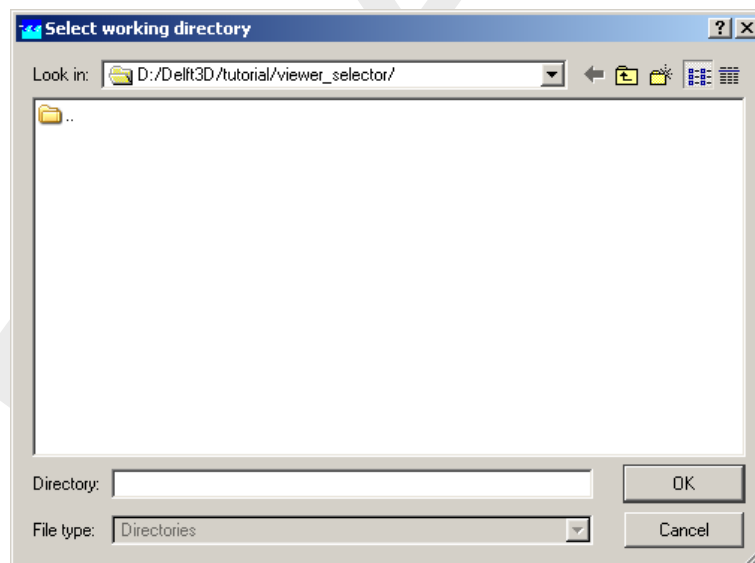


Figure 3.5: *Select working directory window to set the working directory to <viewer_selector>*

Next the **Delft3D Utilities** window is re-displayed, but now the changed current working directory is displayed in the title bar, see [Figure 3.6](#).



Figure 3.6: *Current working directory displayed in title bar*

Now we can select and view NEFIS files.

Hence:

- ◇ Click on *ViewSel.*

The ViewerSelector is loaded and the primary input screen is opened, [Figure 3.7](#).

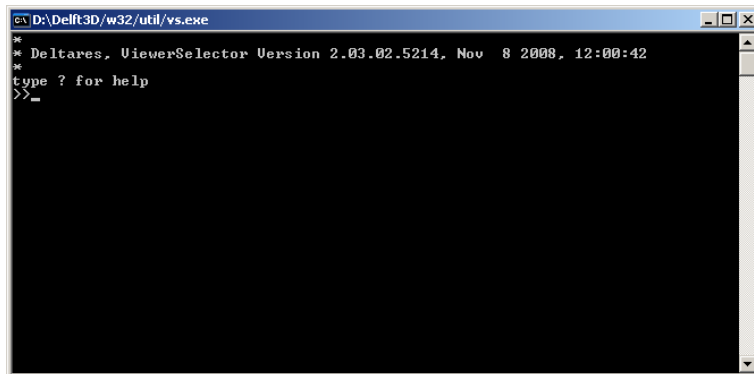


Figure 3.7: Main window of the ViewerSelector

The purpose of the ViewerSelector is to select and view NEFIS files. Selected data can also be exported to an ASCII file.

3.4 Exploring some command options

- ◇ To get an overview of the command line options: enter “?” for help, see [Figure 3.8](#).

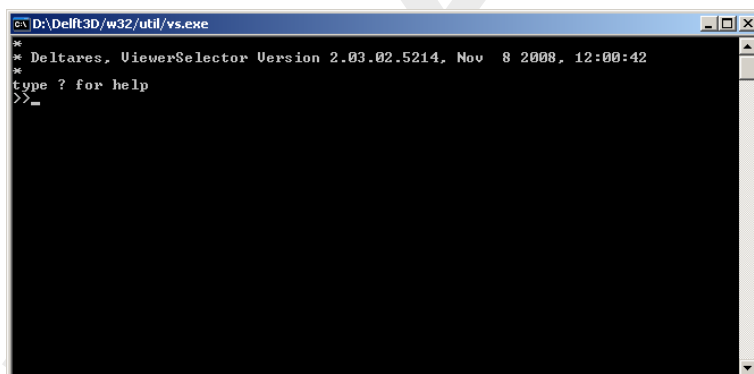


Figure 3.8: Overview of ViewerSelector commands

To open the hydrodynamic map file <trim-f34.*> file:

- ◇ Enter “use trim-f34.dat def trim-f34.def”.

To display an overview of its content:

- ◇ Enter “disp stat”, see [Figure 3.9](#).
- ◇ Press *Enter* or the space-bar several times till you have seen the group ‘map-series’.

The group ‘map-series’ contains the parameters at every grid point for several time points. To get only an overview of the parameters contained in the group ‘map-series’:

- ◇ Enter “disp map-series”, see [Figure 3.10](#).


```

c:\D:\Delft3D\w32\util\vs.exe
*
* Deltares, ViewerSelector Version 2.03.02.5214, Nov  8 2008, 12:00:42
*
type ? for help
>>use trin-f34.dat def trin-f34.def
>>disp stat
datafile:trin-f34.dat
def.file:trin-f34.def

Groupname:map-const      Dimensions:<1>
No attributes
ITDATE      INTEGER * 4      [YYYYMMDD] < 2 >
Initial date (input) & time (default 00:00:00)
TUNIT       REAL * 4      [ S ] < 1 >
Time scale related to seconds
DT           REAL * 4      [ - ] < 1 >
Time step (DT*TUNIT sec)
SIMDAT      CHARACTER* 16 [ - ] < 1 >
Simulation date and time [YYYYMMDD HHMMSS]
SELMAP      CHARACTER* 21 [ - ] < 1 >
Selection flag for field values (2dH, 1dU & 2dU)
NMAX        INTEGER * 4      [ - ] < 1 >
Number of N-grid points
MMAX        INTEGER * 4      [ - ] < 1 >
Number of M-grid points
KMAX        INTEGER * 4      [ - ] < 1 >
Number of layers
LSTCI       INTEGER * 4      [ - ] < 1 >
Number of constituents
LTUR        INTEGER * 4      [ - ] < 1 >
More

```

Figure 3.9: Result from the overview command “disp stat”

```

c:\D:\Delft3D\w32\util\vs.exe
Groupname:map-series      Dimensions:<76>
No attributes
S1          REAL * 4      [ M ] < 22 15 >
Water-level in zeta point
KPU         INTEGER * 4      [ - ] < 22 15 >
Non-active/active in U-point
KPU         INTEGER * 4      [ - ] < 22 15 >
Non-active/active in U-point
U1          REAL * 4      [ M/S ] < 22 15 3 >
U-velocity per layer in U-point (Eulerian)
U1          REAL * 4      [ M/S ] < 22 15 3 >
U-velocity per layer in U-point (Eulerian)
W           REAL * 4      [ M/S ] < 22 15 4 >
W-omega per layer in zeta point
WPHY       REAL * 4      [ M/S ] < 22 15 3 >
W-velocity per layer in zeta point
TAUKSI     REAL * 4      [ N/M2 ] < 22 15 >
Bottom stress in U-point
TAUETA     REAL * 4      [ N/M2 ] < 22 15 >
Bottom stress in U-point
TAUMAX     REAL * 4      [ N/M2 ] < 22 15 >
Tau_max in zeta points (scalar)
VICWV      REAL * 4      [ M2/S ] < 22 15 4 >
Vertical eddy viscosity-3D in zeta point
DICWV      REAL * 4      [ M2/S ] < 22 15 4 >
Vertical eddy diffusivity-3D in zeta point
RICH       REAL * 4      [ - ] < 22 15 4 >
Richardson number
MNKSRG     INTEGER * 4      [ - ] < 7 1 >
More

```

Figure 3.10: Content of the group ‘map-series’

To retrieve the water levels in parameter S1:

- ◇ Enter “let wl = S1 from map-series”.

To see the retrieved water levels on screen (Figure 3.11):

- ◇ Enter “write wl”.

Remark:

- ◇ NPLANE equals 76, indicating at 76 time points water levels are available.
- ◇ Enter “write wl to wl.dat”..



For 76 time points the water levels at every grid point then is written to file <wl.dat>.

```

c:\D:\Delft3D\w32\util\vs.exe
>> let w1 = S1 from map-series
>> write w1
** VARIABLE w1 from map-series(76) ELEMENT S1(22,15) REAL *4
**   NROW      NCOL      NPLANE
**   22         15         76
**
** PLANE: 1
** COLUMN:      1          2          3          4
** ROW
1  0.000000e+000  0.000000e+000  0.000000e+000  0.000000e+000
2  0.000000e+000  0.000000e+000  0.000000e+000  0.000000e+000
3  0.000000e+000  0.000000e+000  0.000000e+000  0.000000e+000
4  0.000000e+000  8.507087e-001  8.461018e-001  8.405338e-001
5  0.000000e+000  8.500484e-001  8.457853e-001  8.425096e-001
6  0.000000e+000  8.523896e-001  8.492353e-001  8.485571e-001
7  0.000000e+000  8.580061e-001  8.563744e-001  8.577996e-001
8  0.000000e+000  8.650435e-001  8.662016e-001  8.687609e-001
9  0.000000e+000  8.741983e-001  8.799886e-001  8.827205e-001
10 0.000000e+000  8.916855e-001  9.021037e-001  9.054683e-001
11 0.000000e+000  9.231140e-001  9.360059e-001  9.397218e-001
12 0.000000e+000  9.687814e-001  9.809686e-001  9.836526e-001
13 0.000000e+000  1.016424e+000  1.024522e+000  1.024207e+000
14 0.000000e+000  1.056685e+000  1.057285e+000  1.053385e+000
15 0.000000e+000  1.081823e+000  1.077207e+000  1.070766e+000
16 0.000000e+000  1.093542e+000  1.087566e+000  1.080205e+000
17 0.000000e+000  1.099102e+000  1.092915e+000  1.085659e+000
18 0.000000e+000  1.101576e+000  1.096047e+000  1.089365e+000
19 0.000000e+000  1.102819e+000  1.098381e+000  1.092190e+000
20 0.000000e+000  1.103524e+000  1.099981e+000  1.094263e+000
21 0.000000e+000  1.102767e+000  1.100369e+000  1.095550e+000

```

Figure 3.11: Overview water levels

3.5 Exiting the ViewerSelector

To exit the ViewerSelector:

- ◇ Enter “exit”, “quit” or “stop”

You will be back in the **Delft3D Utilities** window of the Delft3D-MENU program, [Figure 3.3](#).

- ◇ Click *Return* to return to the main window of Delft3D-MENU, [3.2](#)
- ◇ Click *Exit*.

The window is closed and the control is returned to the desk top or the command line.

In this Getting Started session you have learned to access the ViewerSelector and to load and inspect an existing NEFIS file.

4 General operation

4.1 Environment

The ViewerSelector is or can be implemented on almost every platform, MS-Windows and Linux. No special hardware or software is needed.

For details, concerning implementations and how to configure and start the ViewerSelector see Appendix A.

4.2 User Interface

The ViewerSelector has a command line oriented user interface. This means that it reads all it's commands from *stdin* (normally the keyboard). This implies that commands can be redirected from a file or from a script file. This also means that, for your own special environment, you can build scripts with only a few parameters.

All output, meant to be displayed, is written to *stdout* and (error) messages to *stderr*. If *stdout* and/or *stderr* are connected to a terminal device, which is the default, information will be displayed one page at the time.

4.3 Conventions

ViewerSelector is command line oriented, i.e. it reads all it's commands from *stdin*. Commands consist of reserved words, identifiers, numbers and special characters.

Reserved words and identifiers should be separated by one or more spaces or tabs.

Reserved words

The following table lists the reserved words. Reserved words will be recognised either in lower case, upper case or mixed case.

ALL	AVG	DEF	DELE	DISP
EXEC	EXIT	FROM	HELP	LET
MAX	MEMO	MIN	PAR	RELE
RETN	STAT	STOP	TEKAL	TO
QUIT	USE	WITH	WRITE	

Reserved characters

Also the following characters will be recognised as special.

)	(-	,	;	=	+
-	/	{	}	\	!	?

Number

The ViewerSelector recognises integer and floating point numbers. It uses the C-convention for numbers.

Identifier

An identifier is a string of 1 or more characters, delimited by ' and '. If the string contains only characters from the set [a-zA-Z0-9] and is not equal to a *reserved word* or a *reserved character* or a *number* (see above), the ' and ' can be omitted.

To specify an identifier containing an ', the ' should be escaped by a \.

Examples of correct identifiers:

- ◇ Ident_01
- ◇ 'USE'
- ◇ 'my file'
- ◇ 'my\'file'

filename

Identifier, specifying a file with regard to the filename conventions of the operating system you are using.

datafile

Filename, specifying the name of a NEFIS data file.

definitionfile

Filename, specifying the name of a NEFIS definition file.

elementname

Identifier, specifying a NEFIS element name.

groupname

Identifier, specifying a NEFIS group name.

variable

Identifier, maximum 16 characters, used as a tag for data in memory.

blockname

Identifier, maximum 4 characters, specifying a so-called TEKAL data block name.

value

Value is a *number*.

range

Depending on the number of dimensions an element or group has, you can specify for each dimension a start index, stop index, and step size, separated by commas. The specifications for each dimension must be separated by a semicolon (;). The complete specification must be surrounded by parentheses.

Values may be omitted. The default start index = 1, stop index is the maximum index, found on the NEFIS file and the default step size = 1.

example of range: (1,10;;5,12,2)

meaning from 1 to 10 step 1 in the first dimension, everything from the second dimension and from 5 to 12 step 2 in the third dimension.

process

Identifier.

parameter

Identifier.

DRAFT

DRAFT

5 Commands

This chapter contains the detailed description of the ViewerSelector commands. Commands can be entered when the ViewerSelector prompts for it. The prompt string is ».

The commands are grouped together in a logical way. Appendix B contains an alphabetical list of the commands.

The commands are divided into the following groups:

- ◇ Help
- ◇ NEFIS file commands
- ◇ Data selection and manipulation commands
- ◇ Memory management commands
- ◇ Display commands
- ◇ Write commands
- ◇ Tailor-made process execution commands
- ◇ Shell escape
- ◇ Exit ViewerSelector

Commands must be entered one per line. Lines, starting with a { and ending with a }, will be seen as comment lines.

Notational conventions

You can type commands and parameters in either uppercase or lowercase. The following notational conventions are used.

SAMPLE	Bold letters indicate a specific term to be used literally.
<i>italic</i>	Words in italics mean that you have to provide the actual value.
	Separates two mutual exclusive choices. Type only one of these choices.
[]	Indicates an item that is optional. To include the optional information, type only the information, not the brackets.
...	Indicates that the previous parameter or switch can be repeated several times in a command. Type only the information, not the ... itself.

5.1 Help command

HELP |?

A brief summary of the allowed commands is printed to *stderr*.

5.2 NEFIS file command

USE datafile DEF definitionfile

Open a NEFIS data file and it's associated definition file. Only one set of files can be open. An already opened pair of NEFIS files will be closed before opening the new set of files. **USE** Close an open set of NEFIS files.

5.3 Display commands

DISP MEMO

Display information about variables, currently in memory.

DISP STAT [TO filename]

Display status and meta data of current open NEFIS-file. If you specify a filename, status information will be appended to that file.

DISP *groupname*

Display meta data of the specified *groupname*.

5.4 Data selection and manipulation commands

The following commands describe how data can be read from file to memory and how to perform some simple operations on this data. The values, read from file, will be tagged with a name, called the variable name. You must use this name in all commands when you want to refer to this data. It is also possible to create a simple floating point memory variable with an assignment statement.

LET *variable* = *value*

This command creates a memory variable with the name *variable* and will be assigned the value *value*. Value can either be an integer or float. The created memory variable will always be a float (i. e. REAL*4).

LET *variable* = *variable* + |- |* |/*variable*|*value*

This command allows you to perform some basic operations (add, subtract, multiply, divide) with existing memory variables. It creates a new variable.

This command is limited to operations on data types: double, float, int and short (in Fortran: REAL*8, REAL*4, INT*4 and INT*2). The return value is always a float (i.e. REAL*4).

The data types of the variables on the right hand side of the equal sign need not be the same. This means you can add a float with an integer etc. If you specify on the right hand side two variables, then they must have the same structure, meaning the same element and group dimensions.

This function does not check for floating underflow or overflow.

Dividing by zero will give a result: 999.999.

LET *variable* = MAX |MIN |AVG *variable* [*value*]

This command calculates the maximum, minimum or average value of the variable on the right hand side. It creates a new memory variable, containing this calculated value.

This command is limited to operations on data types: double, float, int and short (in Fortran: REAL*8, REAL*4, INT*4 and INT*2). The return value is always a float (i.e. REAL*4).

Optionally you may specify a value. This value is used as exclude value during the operation. (Example: 999.999 excludes the results from divisions by zero)

LET *variable* = *elementname* [*range*] FROM *groupname* [*range*]

This command reads the data of the specified element of the specified group from the current open set of NEFIS files to memory. If no error occurred during the data retrieval, this data is tagged with the name *variable*.

With range you can limit the amount of data to be read. For a definition of the syntax of range, see Section 4.3.

5.5 Memory management commands

RELE ALL

Release all memory variables. This also frees the occupied space.

RELE *variable*

Release specified variable from memory.

5.6 Write command

WRITE *variable* ...

Write contents of specified variables to *stdout*. *Stdout* is by default connected to your terminal, so output will be displayed on your screen, page by page.

Character variables will be truncated to maximum 16 characters.

You may specify 1 to 40 variables. If you specify more than 1 variable, these variables must have the same structure (not necessarily the same type).

If the variables have more than 3 dimensions, the write command will not write the contents to a file. If you still want to write such complex data structures to a file you have to write your own processes and execute such a process with the EXEC-command.

WRITE *variable* ... TO filename [blockname]

Write contents of ASCII file with a so called TEKAL-layout.

If the file already exists, data will be appended.

You may specify 1 to 10 variables. If you specify more than 1 variable, they must have the same structure (not necessarily the same type).

You may optionally specify a block name of maximum 4 characters. The default block name = 'XXXX'.

5.7 Tailor-made process execution commands

With these commands you can execute another process without leaving the ViewerSelector.

You can use this command to start tailor-made processes that use data provided by the ViewerSelector, return data to the ViewerSelector or a combination of both.

But you can also use these commands to start any operating system command, program or shell script or DOS batch file. In the latter case you better use the Shell escape command.

If you start a process this way in the DOS-environment, it means that the ViewerSelector starts the execution of a child process, waits until this child is finished and then resumes.

Processes, which don not need variables from the ViewerSelector and/or do not return data may be executed in background (Linux and UNIX only).

EXEC *process* [RETN]

Execute specified process. If you specify the optional keyword RETN, the ViewerSelector will try to read new memory variables from a temporarily file, written by the process (see ViewerSelector Programmers Manual).

EXEC *process* WITH *variable* ... [RETN]

This command opens a pipe between the ViewerSelector and the specified process and writes the contents of the variables to that pipe. The specified process must be suitable to read these variables from the pipe (see also ViewerSelector Programmers Manual).

If you do not specify the keyword RETN, the ViewerSelector assumes that no information will return to the ViewerSelector.

If you specify the keyword RETN, the ViewerSelector will try to read new memory variables from a temporarily file, created and written by the process in a special way (see ViewerSelector Programmers Manual).

EXEC *process* PAR *parameter* ... [RETN]

Execute specified process with parameters.

If you specify the keyword RETN, the ViewerSelector assumes that the process will return new variables (see previous command).

EXEC *process* WITH *variable* ... PAR *parameter* ... [RETN]

This command is a combination of the two commands above.

5.8 Shell escape command

! *identifier*

Everything you specify after the shell escape character (!), will be send to a new shell and executed immediately.

5.9 Exit ViewerSelector

EXIT |QUIT |STOP

Entering this command will immediately stop the ViewerSelector.

6 Error messages

All error messages are displayed to *stderr*. The error messages are self-explaining. List of error messages:

```
Out of memory
Error reading def.file (NEFIS error <number>)
Error opening def.file
Error opening datafile
Error opening files
Element <elementname> does not exist
Error reading datafile (NEFIS error <number>)
Variable already exists
Group does not exist
Element indices not correct
Group indices not correct
Variable <variable name> does not exist
Unable to open file <filename>
Variable <variable name> and variable <variable name> have different structures
Variable <variable name> not created
Variable type incorrect for function
Variables have different structure
Process <process name> cannot be opened
Unknown errorcode <number>
```

DRAFT

7 Tutorial

This chapter shows some sample ViewerSelector sessions. Commands, used in the sessions are numbered in the right margin. At the end of each session the commands are explained.

7.1 Example 1

In this first example we retrieve some floating point data from a NEFIS file and write it to an ASCII file. Follow the instructions in Chapter 3 to start the ViewerSelector.

```
>>use test.dat def test.def (1)
>>disp stat \hfill (2)
datafile:test.dat
def.file:test.def
Groupname:GrpName Dimensions:(3,5,7)
No attributes
Value REAL * 4
>>LET data = Value FROM GrpName (3)
>>disp memo (4)
VARIABLE data from GrpName(3,5,7) ELEMENT Value\ REAL *4
>>write data to out (5)
>>! more out (6)
** VARIABLE data from GrpName(3,5,7) ELEMENT Value REAL *4
** NROW NCOL NPLANE
XXXX
      21      5      7
1.000000e+00 4.000000e+00 7.000000e+00 1.000000e+01 1.300000e+01
2.000000e+00 5.000000e+00 8.000000e+00 1.100000e+01 1.400000e+01
3.000000e+00 6.000000e+00 9.000000e+00 1.200000e+01 1.500000e+01
1.600000e+01 1.900000e+01 2.200000e+01 2.500000e+01 2.800000e+01
1.700000e+01 2.000000e+01 2.300000e+01 2.600000e+01 2.900000e+01
1.800000e+01 2.100000e+01 2.400000e+01 2.700000e+01 3.000000e+01
3.100000e+01 3.400000e+01 3.700000e+01 4.000000e+01 4.300000e+01
3.200000e+01 3.500000e+01 3.800000e+01 4.100000e+01 4.400000e+01
3.300000e+01 3.600000e+01 3.900000e+01 4.200000e+01 4.500000e+01
4.600000e+01 4.900000e+01 5.200000e+01 5.500000e+01 5.800000e+01
4.700000e+01 5.000000e+01 5.300000e+01 5.600000e+01 5.900000e+01
4.800000e+01 5.100000e+01 5.400000e+01 5.700000e+01 6.000000e+01
6.100000e+01 6.400000e+01 6.700000e+01 7.000000e+01 7.300000e+01
6.200000e+01 6.500000e+01 6.800000e+01 7.100000e+01 7.400000e+01
6.300000e+01 6.600000e+01 6.900000e+01 7.200000e+01 7.500000e+01
7.600000e+01 7.900000e+01 8.200000e+01 8.500000e+01 8.800000e+01
7.700000e+01 8.000000e+01 8.300000e+01 8.600000e+01 8.900000e+01
7.800000e+01 8.100000e+01 8.400000e+01 8.700000e+01 9.000000e+01
9.100000e+01 9.400000e+01 9.700000e+01 1.000000e+02 1.030000e+02
9.200000e+01 9.500000e+01 9.800000e+01 1.010000e+02 1.040000e+02
9.300000e+01 9.600000e+01 9.900000e+01 1.020000e+02 1.050000e+02
>>exit \hfill (7)
```

Explanation

- 1 This command opens the NEFIS files <test.dat> and <test.def>. If your file is not in the current directory, but let's assume in the parent directory, the command would be:
use './test.dat' def './test.def'
- 2 Display the status. First you see the names of the currently open NEFIS files, followed by a list of all data groups, found on the data file (in this case only one: GrpName).
- 3 Read the data from the NEFIS file and tag it with the name *data*.
- 4 With this command we can see what we have in memory, in this case only the variable *data*.
- 5 This command writes the contents of variable *data* to file out. If out already exists, the information will be appended.
- 6 With a shell escape we show the contents of file out.
- 7 Exit the ViewerSelector.

7.2 Example 2

In this second example we do some manipulation with the retrieved data. We will not retrieve the complete data set, but just a part.

```

>>use t.dat def t.def (1)
>>let p1 = Value from GrpName (;;1,1) (2)
>>let p2 = Value from GrpName (;;2,2) (3)
>>let mul = p1 * p2 (4)
>>let 'min' = min mul (5)
>>let 'max' = max mul (6)
>>let 'avg' = avg mul (7)
>>write 'min' 'max' 'avg' (8)
** VARIABLE min from --Derived--(1) ELEMENT --Minimum--(1) REAL *4
** VARIABLE max from --Derived--(1) ELEMENT --Maximum--(1) REAL *4
** VARIABLE avg from --Derived--(1) ELEMENT --Average--(1) REAL *4
**   NROW      NCOL
**     1         3
** COLUMN:      1         2         3
**   ROW
**     1  1.600000e+01  4.500000e+02  2.026667e+02
>>write p1 p2 mul (9)
** VARIABLE p1 from GrpName(3,5) ELEMENT Value(1) REAL *4
** VARIABLE p2 from GrpName(3,5) ELEMENT Value(1) REAL *4
** VARIABLE mul from GrpName(3,5) ELEMENT *(1) REAL *4
**   NROW      NCOL
**    15         3
** COLUMN:      1         2         3
**   ROW
**     1  1.000000e+00  1.600000e+01  1.600000e+01
**     2  2.000000e+00  1.700000e+01  3.400000e+01
**     3  3.000000e+00  1.800000e+01  5.400000e+01
**     4  4.000000e+00  1.900000e+01  7.600000e+01
**     5  5.000000e+00  2.000000e+01  1.000000e+02
**     6  6.000000e+00  2.100000e+01  1.260000e+02
**     7  7.000000e+00  2.200000e+01  1.540000e+02
**     8  8.000000e+00  2.300000e+01  1.840000e+02
**     9  9.000000e+00  2.400000e+01  2.160000e+02
**    10  1.000000e+01  2.500000e+01  2.500000e+02
**    11  1.100000e+01  2.600000e+01  2.860000e+02
**    12  1.200000e+01  2.700000e+01  3.240000e+02
**    13  1.300000e+01  2.800000e+01  3.640000e+02
**    14  1.400000e+01  2.900000e+01  4.060000e+02
**    15  1.500000e+01  3.000000e+01  4.500000e+02
>>exit (10)

```

Explanation

- 1 See example 1
- 2, 3 Read data from file, but not the complete group. We want to read only one plane.
- 4 Multiply $p1$ with $p2$ and tag result with mul .
- 5, 6, 7 Calculate minimum, maximum and average values of mul and tag them with the names min , max and avg . These tags must be specified between ' and ', because they are the same as a reserved word.
- 8 Display minimum, maximum and average.
- 9 Display $p1$, $p2$ and mul .
- 10 Exit ViewerSelector

7.3 Example 3

In this example we use the ViewerSelector from within a script file with three parameters. The actions we take are the same as in the previous example, but the plane numbers are variable and the result is written to a file. First we present you the script file:

```
#!/bin/sh
# first test the number of parameters
if [ $# -ne 3 ] ; then
    echo "usage: $0 plane1 plane2 file" >&2
    exit 1
fi
# call the Viewer/Selector using a so-called here-file.
# all information, written to stderr, is now redirected to file error.
vs <<++++ 2>error
use t.dat def t.def
{ $1 and $2 are the plane numbers }
let p1 = Value from GrpName (;$1,$1)
let p2 = Value from GrpName (;$2,$2)
let mul = p1 * p2
let 'min' = min mul
let 'max' = max mul
let 'avg' = avg mul
{ $3 is the name of output file, remove this file }
!rm $3
{ write info to file using so-called Tekal-block names }
write 'min' 'max' 'avg' to $3 B001
write p1 p2 mul to $3 B002
quit
++++
```

Let's execute this script, named ex3.

```
./ex3 1 2 out
```

The result is in file out, so lets examine out.

```
more out
```

```
** VARIABLE min from --Derived--(1) ELEMENT --Minimum--(1) REAL *4
** VARIABLE max from --Derived--(1) ELEMENT --Maximum--(1) REAL *4
** VARIABLE avg from --Derived--(1) ELEMENT --Average--(1) REAL *4
**   NROW      NCOL
      1         3
B001
  4.600000e+01  9.000000e+02  4.426666e+02
** VARIABLE p1 from GrpName(3,5) ELEMENT Value(1) REAL *4
** VARIABLE p2 from GrpName(3,5) ELEMENT Value(1) REAL *4
** VARIABLE mul from --Derived--(3,5) ELEMENT *(1) REAL *4
**   NROW      NCOL
      15         3
B002
  1.000000e+00  4.600000e+01  4.600000e+01
  2.000000e+00  4.700000e+01  9.400000e+01
  3.000000e+00  4.800000e+01  1.440000e+02
  4.000000e+00  4.900000e+01  1.960000e+02
  5.000000e+00  5.000000e+01  2.500000e+02
  6.000000e+00  5.100000e+01  3.060000e+02
  7.000000e+00  5.200000e+01  3.640000e+02
  8.000000e+00  5.300000e+01  4.240000e+02
  9.000000e+00  5.400000e+01  4.860000e+02
  1.000000e+01  5.500000e+01  5.500000e+02
  1.100000e+01  5.600000e+01  6.160000e+02
  1.200000e+01  5.700000e+01  6.840000e+02
  1.300000e+01  5.800000e+01  7.540000e+02
  1.400000e+01  5.900000e+01  8.260000e+02
```

```
1.500000e+01 6.000000e+01 9.000000e+02
```

You will notice that the result is the same as in the previous example.

This example showed you, we hope, the possibility to make dedicated script files for every day actions.

7.4 Example 4

In this example we use the EXEC-command to generate values, which are returned to the ViewerSelector. The EXEC command is used to start a process which draws a simple XY-graph.

The script-file looks like this:

```
\#!/bin/sh
\# first test the number of parameters
if [ \${#} -ne 1 ] ; then
    echo "usage: \${0} station-number" >\&2
    exit 1
fi
vs <<++++ 2>/dev/null (1)
use tr.dat def tr.def (2)
let y = ZWL (\$1,\$1) from his-series (3)
exec basfun par x '85' retn (4)
let xmin = 0 (5)
let xmax = 84 (6)
let ymin = min y (7)
let ymax = max y (8)
exec xy-graph with xmin xmax ymin ymax x y par "'Station \$1"' (9)
quit (10)
++++
```

Explanation

- 1 Start ViewerSelector, using the here-file mechanism. All output to *stderr* is redirected to `</dev/null>`, which means nowhere.
- 2 The file we are using, is a FLOW file.
- 3 Select here some water levels, 85 floats.
- 4 Execute a process, outside the ViewerSelector. Before we can use such a process, we should know how to use it. This process should be called with 2 parameters, the first is a text, the second a number. This process generates a one dimensional array with a length as specified in the second parameter, in this case 85. At end it will contain the values from 1 to 85. It returns this data back to the ViewerSelector. The first parameter is the tag for the returned data. This process is used the generate a base for the XY-graph.
- 5, 6, 7, 8 Calculate minimum and maximum values.
- 9 Execute process XY-graph. This process needs 6 data values and 1 parameter. It will generate is simple XY graph.
- 10 Exit the ViewerSelector.

Let's execute this script, named ex4.

```
./ex4 1
```

So now we get an XY-graph of the water levels of station 1.

References

ViewerSelector, 1993. *ViewerSelector Programmer's Manual*. Deltares, 1.00 ed.

DRAFT

DRAFT

A Appendix

A.1 Distribution and installation notes

A.1.1 Environment

In the Linux environment the ViewerSelector uses two environment variables, TMPDIR and PAGER.

TMPDIR is used to specify the location of temporarily files. If this variable is not set these files are located in a system default place (normally </tmp>).

With PAGER you can specify the filter program to be used for displaying information to your screen. You can use filters like more, pg, etc. In the MS-Windows environment the ViewerSelector also uses two environment variables, TMP and PAGER. The meaning of these variables is the same as is in the Linux environment. For PAGER you can use MORE.EXE.

A.1.2 Starting the ViewerSelector

Once you have located the ViewerSelector and have set your environment variables, you can start the program by typing:

```
vs
```

If everything is okay you see the following message on your screen:

```
type ? for help  
>>
```

now the installation is ready to use.

A.2 Command summary

Command Summary

```
DISP MEMO\textbar STAT [TO \textit{filename}]\textbar \textit{groupname}
EXEC \textit{process} [WITH \textit{variable ...}] [PAR \textit{parameter ...}] [RETN]
EXIT\textbar QUIT\textbar STOP
HELP\textbar ?
LET \textit{variable} = \textit{value}
LET \textit{variable} = +\textbar -\textbar *\textbar / \textit{variable}\textbar value}
LET \textit{variable} = MAX\textbar MIN\textbar AVG \textit{variable} [\textit{value}]
LET \textit{variable} = \textit{elementname} [\textit{range}] FROM \textit{groupname} [\textit{range}]
QUIT\textbar EXIT\textbar STOP
RELE ALL\textbar \textit{variable}
STOP\textbar EXIT\textbar QUIT
USE [\textit{datafile} DEF \textit{definitionfile}]
WRITE \textit{variable ...} [TO \textit{filename} [\textit{blockname}]]
```


DRAFT



Deltares **systems**

PO Box 177
2600 MH Delft
Rotterdamseweg 185
2629 HD Delft
The Netherlands

+31 (0)88 335 81 88
sales@deltaressystem.nl
www.deltaressystem.nl